

1 What is claimed is:

1 1. A method for mapping components of an XML schema using a program,
2 comprising:

3 a. uniquely mapping XML schema components with a conversion language;
4 and

5 b. uniquely naming components of the conversion language based on names of
6 the XML schema components.

1 2. The method of claim 1, wherein the conversion language is Java, and step a.
2 further comprises (1) mapping each XML schema element and type to a Java
3 component and (2) uniquely identifying each XML schema element and type
4 within a set of all distinct XML schema.

1 3. The method of claim 1, wherein the conversion language is Java, and step b.
2 further comprises, for each different XML schema element and type,
3 generating a unique Java component name.

1 4. The method of claim 3, wherein the step of generating each said unique
2 Java component name comprises generating each name so that each said name
3 substantially adheres to Java naming standards, and so that each said name
4 remains the same in subsequent mappings of XML schema components when
5 an XML schema component label on which said name is based remains the
6 same.

1 5. The method of claim 1:

2 wherein the conversion language is Java; step a. further comprises (1)
3 mapping each XML schema element, type and attribute to a Java component
4 and (2) uniquely identifying each XML schema element, type and attribute
5 within a set of all distinct XML schema; and step b. further comprises, for each
6 different XML schema element, type and attribute name, generating a unique
7 Java component name; and

8 the method further comprises generating a reusable definition object operable
9 in converting information between an XML object associated with the XML
10 schema and a Java object.

1 6. The method of claim 5, wherein step b. further comprises at least one of the
2 group of (a) hashing the name of each element having the same name and type
3 as another element to form an element name hash code part of a Java member
4 variable, (b) hashing the name of each attribute having the same name as
5 another attribute to form an attribute name hash code part of a further Java
6 member variable, (c) hashing a QName for each complex type to form a
7 complex type name hash code part of a Java class name, (d) hashing each
8 LongName name to form a LongName hash code part of a truncated Java
9 name, and (e) hashing a concatenated string of all component names of an
10 anonymous complex type component to form an anonymous complex type hash
11 code part of a further Java class name.

1 7. The method of claim 6, further comprising appending a suffix to a
2 generated Java component name based on a first XML schema component
3 name when the generated Java component name is identical to a previously
4 generated Java component name based on a second XML schema component
5 name different from the first XML schema component name.

1 8. An information handling system comprising a processor and an object
2 definition tool for generating an object operable in mapping components of an
3 XML schema, the object definition tool comprising plural instructions and the
4 processor is operably configured to execute said plural instructions, the plural
5 instructions comprising:

6 a. mapping instructions operable for uniquely mapping XML schema
7 components with a conversion language; and

8 b. naming instructions operable for uniquely naming components of the
9 conversion language based on names of the XML schema components.

1 9. The information handling system of claim 8, wherein the conversion
2 language is Java, and the mapping instructions comprise instructions
3 configured to (1) map each XML schema element and type to a Java
4 component and (2) uniquely identify each XML schema element and type
5 within a set of all distinct XML schema.

1 10. The information handling system of claim 8, wherein the conversion
2 language is Java, and the naming instructions further comprise instructions,
3 for each different XML schema element and type, configured to generate a
4 unique Java component name.

1 11. The information handling system of claim 10, wherein naming
2 instructions comprise yet further instructions configured to generate each
3 name so that each said name substantially adheres to Java naming standards,
4 and so that each said name remains the same in subsequent mappings of XML
5 schema components when an XML schema component label on which said
6 name is based remains the same.

1 12. The information handling system of claim 8:
2 wherein the conversion language is Java; the mapping instructions comprise
3 instructions configured to (1) map each XML schema element and type to a
4 Java component and (2) uniquely identify each XML schema element and type
5 within a set of all distinct XML schema; and the naming instructions further
6 comprise additional instructions configured to generate, for each different
7 XML schema element and type, a unique Java component name;
8 the system further comprising object definition instructions configured to
9 generate a reusable definition object operable in converting information
10 between an XML object associated with the XML schema and a Java object.

1 13. The system of claim 12, wherein the naming instructions further comprise
2 hashing instructions configured to perform at least one of the group of
3 operations (a) hashing the name of each element having the same name and

4 type as another element to form an element name hash code part of a Java
5 member variable, (b) hashing the name of each attribute having the same
6 name as another attribute to form an attribute name hash code part of a
7 further Java member variable, (c) hashing a QName for each complex type to
8 form a complex type name hash code part of a Java class name, (d) hashing
9 each LongName name to form a LongName hash code part of a truncated Java
10 name, and (e) hashing a concatenated string of all component names of an
11 anonymous complex type component to form an anonymous complex type hash
12 code part of a further Java class name.

1 14. The system of claim 13, further comprising appending instructions
2 operable for appending a suffix to a generated Java component name based on
3 a first XML schema component name when the generated Java component
4 name is identical to a previously generated Java component name based on a
5 second XML schema component name different from the first XML schema
6 component name.

1 15. The system of claim 12, further comprising an adapter configured to
2 operably convert XML objects using the reusable definition object to converted
3 XML objects, and an application and computer operable together and
4 configured to receive and perform operations on the converted XML objects.

1 16. A program product in a signal bearing medium executable by a device for
2 generating an object operable in mapping components of an XML schema, the
3 product comprising:

4 a. mapping instructions operable for uniquely mapping XML schema
5 components with a conversion language; and

6 b. naming instructions operable for uniquely naming components of the
7 conversion language based on names of the XML schema components.

1 17. The program product of claim 16, wherein the conversion language is
2 Java, and the mapping instructions comprise instructions configured to (1)

3 map each XML schema element and type to a Java component and (2)
4 uniquely identify each XML schema element and type within a set of all
5 distinct XML schema.

1 18. The program product of claim 16, wherein the conversion language is
2 Java, and the naming instructions further comprise instructions, for each
3 different XML schema element and type, configured to generate a unique Java
4 component name.

1 19. The program product of claim 18, wherein naming instructions comprise
2 yet further instructions configured to generate each name so that each said
3 name substantially adheres to Java naming standards, and so that each said
4 name remains the same in subsequent mappings of XML schema components
5 when an XML schema component label on which said name is based remains
6 the same.

1 20. The program product of claim 16:
2 wherein the conversion language is Java; the mapping instructions comprise
3 instructions configured to (1) map each XML schema element and type to a
4 Java component and (2) uniquely identify each XML schema element and type
5 within a set of all distinct XML schema; and the naming instructions further
6 comprise additional instructions configured to generate, for each different
7 XML schema element and type, a unique Java component name;
8 the product further comprising object definition instructions configured to
9 generate a reusable definition object operable in converting information
10 between an XML object associated with the XML schema and a Java object.

1 21. The program product of claim 20, wherein the naming instructions further
2 comprise hashing instructions configured to perform at least one of the group
3 of operations (a) hashing the name of each element having the same name and
4 type as another element to form an element name hash code part of a Java
5 member variable, (b) hashing the name of each attribute having the same

6 name as another attribute to form an attribute name hash code part of a
7 further Java member variable, (c) hashing a QName for each complex type to
8 form a complex type name hash code part of a Java class name, (d) hashing
9 each LongName name to form a LongName hash code part of a truncated Java
10 name, and (e) hashing a concatenated string of all component names of an
11 anonymous complex type component to form an anonymous complex type hash
12 code part of a further Java class name.

1 22. The program product of claim 21, further comprising appending
2 instructions operable for appending a suffix to a generated Java component
3 name based on a first XML schema component name when the generated Java
4 component name is identical to a previously generated Java component name
5 based on a second XML schema component name different from the first XML
6 schema component name.

1 23. The program product of claim 22, further comprising adapter instructions
2 configured to operably convert XML objects using the reusable definition object
3 to converted XML objects.